

NAG C Library Function Document

nag_dtptrs (f07uec)

1 Purpose

nag_dtptrs (f07uec) solves a real triangular system of linear equations with multiple right-hand sides, $AX = B$ or $A^T X = B$, using packed storage.

2 Specification

```
#include <nag.h>
#include <nagf07.h>

void nag_dtptrs (Nag_OrderType order, Nag_UploType uplo, Nag_TransType trans,
                 Nag_DiagType diag, Integer n, Integer nrhs, const double ap[], double b[],
                 Integer pdb, NagError *fail)
```

3 Description

nag_dtptrs (f07uec) solves a real triangular system of linear equations $AX = B$ or $A^T X = B$ using packed storage.

4 References

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

Higham N J (1989) The accuracy of solutions to triangular systems *SIAM J. Numer. Anal.* **26** 1252–1265

5 Arguments

- 1: **order** – Nag_OrderType *Input*
On entry: the **order** argument specifies the two-dimensional storage scheme being used, i.e., row-major ordering or column-major ordering. C language defined storage is specified by **order** = **Nag_RowMajor**. See Section 2.2.1.4 of the Essential Introduction for a more detailed explanation of the use of this argument.
Constraint: **order** = **Nag_RowMajor** or **Nag_ColMajor**.
- 2: **uplo** – Nag_UploType *Input*
On entry: indicates whether A is upper or lower triangular.
uplo = **Nag_Upper**
 A is upper triangular.
uplo = **Nag_Lower**
 A is lower triangular.
Constraint: **uplo** = **Nag_Upper** or **Nag_Lower**.
- 3: **trans** – Nag_TransType *Input*
On entry: indicates the form of the equations.
trans = **Nag_NoTrans**
The equations are of the form $AX = B$.

trans = Nag_Trans or Nag_ConjTrans

The equations are of the form $A^T X = B$.

Constraint: **trans** = Nag_NoTrans, Nag_Trans or Nag_ConjTrans.

4: **diag** – Nag_DiagType *Input*

On entry: indicates whether A is a non-unit or unit triangular matrix.

diag = Nag_NonUnitDiag

A is a non-unit triangular matrix.

diag = Nag_UnitDiag

A is a unit triangular matrix; the diagonal elements are not referenced and are assumed to be 1.

Constraint: **diag** = Nag_NonUnitDiag or Nag_UnitDiag.

5: **n** – Integer *Input*

On entry: n , the order of the matrix A .

Constraint: $n \geq 0$.

6: **nrhs** – Integer *Input*

On entry: r , the number of right-hand sides.

Constraint: **nrhs** ≥ 0 .

7: **ap**[*dim*] – const double *Input*

Note: the dimension, *dim*, of the array **ap** must be at least $\max(1, n \times (n + 1)/2)$.

On entry: the n by n triangular matrix A , packed by rows or columns. The storage of elements a_{ij} depends on the **order** and **uplo** arguments as follows:

if **order** = Nag_ColMajor and **uplo** = Nag_Upper,
 a_{ij} is stored in **ap**[($j - 1$) $\times$ $j/2 + i - 1$], for $i \leq j$;
 if **order** = Nag_ColMajor and **uplo** = Nag_Lower,
 a_{ij} is stored in **ap**[($2n - j$) $\times$ ($j - 1$)/2 + $i - 1$], for $i \geq j$;
 if **order** = Nag_RowMajor and **uplo** = Nag_Upper,
 a_{ij} is stored in **ap**[($2n - i$) $\times$ ($i - 1$)/2 + $j - 1$], for $i \leq j$;
 if **order** = Nag_RowMajor and **uplo** = Nag_Lower,
 a_{ij} is stored in **ap**[($i - 1$) $\times$ $i/2 + j - 1$], for $i \geq j$.

8: **b**[*dim*] – double *Input/Output*

Note: the dimension, *dim*, of the array **b** must be at least

$\max(1, \mathbf{pdb} \times \mathbf{nrhs})$ when **order** = Nag_ColMajor;
 $\max(1, \mathbf{pdb} \times n)$ when **order** = Nag_RowMajor.

If **order** = Nag_ColMajor, the (i, j)th element of the matrix B is stored in **b**[($j - 1$) $\times$ **pdb** + $i - 1$].

If **order** = Nag_RowMajor, the (i, j)th element of the matrix B is stored in **b**[($i - 1$) $\times$ **pdb** + $j - 1$].

On entry: the n by r right-hand side matrix B .

On exit: the n by r solution matrix X .

9: **pdb** – Integer *Input*

On entry: the stride separating matrix row or column elements (depending on the value of **order**) in the array **b**.

Constraints:

if **order** = **Nag_ColMajor**, **pdb** $\geq \max(1, \mathbf{n})$;
 if **order** = **Nag_RowMajor**, **pdb** $\geq \max(1, \mathbf{nrhs})$.

10: **fail** – NagError *

Input/Output

The NAG error argument (see Section 2.6 of the Essential Introduction).

6 Error Indicators and Warnings

NE_ALLOC_FAIL

Dynamic memory allocation failed.

NE_BAD_PARAM

On entry, argument $\langle value \rangle$ had an illegal value.

NE_INT

On entry, **n** = $\langle value \rangle$.

Constraint: **n** ≥ 0 .

On entry, **nrhs** = $\langle value \rangle$.

Constraint: **nrhs** ≥ 0 .

On entry, **pdb** = $\langle value \rangle$.

Constraint: **pdb** > 0 .

NE_INT_2

On entry, **pdb** = $\langle value \rangle$, **n** = $\langle value \rangle$.

Constraint: **pdb** $\geq \max(1, \mathbf{n})$.

On entry, **pdb** = $\langle value \rangle$, **nrhs** = $\langle value \rangle$.

Constraint: **pdb** $\geq \max(1, \mathbf{nrhs})$.

NE_INTERNAL_ERROR

An internal error has occurred in this function. Check the function call and any array sizes. If the call is correct then please consult NAG for assistance.

NE_SINGULAR

The matrix A is singular.

7 Accuracy

The solutions of triangular systems of equations are usually computed to high accuracy. See Higham (1989).

For each right-hand side vector b , the computed solution x is the exact solution of a perturbed system of equations $(A + E)x = b$, where

$$|E| \leq c(n)\epsilon|A|,$$

$c(n)$ is a modest linear function of n , and ϵ is the *machine precision*.

If $\hat{x}$ is the true solution, then the computed solution x satisfies a forward error bound of the form

$$\frac{\|x - \hat{x}\|_{\infty}}{\|x\|_{\infty}} \leq c(n) \operatorname{cond}(A, x)\epsilon, \quad \text{provided} \quad c(n) \operatorname{cond}(A, x)\epsilon < 1,$$

where $\operatorname{cond}(A, x) = \| |A^{-1}| |A| |x| \|_{\infty} / \|x\|_{\infty}$.

Note that $\text{cond}(A, x) \leq \text{cond}(A) = \|A^{-1}\| \|A\|_{\infty} \leq \kappa_{\infty}(A)$; $\text{cond}(A, x)$ can be much smaller than $\text{cond}(A)$ and it is also possible for $\text{cond}(A^T)$ to be much larger (or smaller) than $\text{cond}(A)$.

Forward and backward error bounds can be computed by calling `nag_dtpfrs` (f07uhc), and an estimate for $\kappa_{\infty}(A)$ can be obtained by calling `nag_dtpcon` (f07ugc) with **norm** = **Nag_InfNorm**.

8 Further Comments

The total number of floating-point operations is approximately $n^2 r$.

The complex analogue of this function is `nag_ztptrs` (f07usc).

9 Example

To solve the system of equations $AX = B$, where

$$A = \begin{pmatrix} 4.30 & 0.00 & 0.00 & 0.00 \\ -3.96 & -4.87 & 0.00 & 0.00 \\ 0.40 & 0.31 & -8.02 & 0.00 \\ -0.27 & 0.07 & -5.95 & 0.12 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} -12.90 & -21.50 \\ 16.75 & 14.93 \\ -17.55 & 6.33 \\ -11.04 & 8.09 \end{pmatrix},$$

using packed storage for A .

9.1 Program Text

```
/* nag_dtptrs (f07uec) Example Program.
 *
 * Copyright 2001 Numerical Algorithms Group.
 *
 * Mark 7, 2001.
 */

#include <stdio.h>
#include <nag.h>
#include <nag_stdlib.h>
#include <nagf07.h>
#include <nagx04.h>

int main(void)
{
    /* Scalars */
    Integer ap_len, i, j, n, nrhs, pdb;
    Integer exit_status=0;
    Nag_UploType uplo_enum;

    NagError fail;
    Nag_OrderType order;
    /* Arrays */
    char uplo[2];
    double *ap=0, *b=0;

#ifdef NAG_COLUMN_MAJOR
#define A_UPPER(I,J) ap[J*(J-1)/2 + I - 1]
#define A_LOWER(I,J) ap[(2*n-J)*(J-1)/2 + I - 1]
#define B(I,J) b[(J-1)*pdb + I - 1]
    order = Nag_ColMajor;
#else
#define A_LOWER(I,J) ap[I*(I-1)/2 + J - 1]
#define A_UPPER(I,J) ap[(2*n-I)*(I-1)/2 + J - 1]
#define B(I,J) b[(I-1)*pdb + J - 1]
    order = Nag_RowMajor;
#endif

    INIT_FAIL(fail);
    Vprintf("nag_dtptrs (f07uec) Example Program Results\n\n");
```

```

/* Skip heading in data file */
Vscanf("%*[\n] ");
Vscanf("%ld%ld%*[\n] ", &n, &nrhs);
ap_len = n*(n+1)/2;
#ifdef NAG_COLUMN_MAJOR
    pdb = n;
#else
    pdb = nrhs;
#endif

/* Allocate memory */
if ( !(ap = NAG_ALLOC(ap_len, double)) ||
     !(b = NAG_ALLOC(n * nrhs, double)) )
{
    Vprintf("Allocation failure\n");
    exit_status = -1;
    goto END;
}

/* Read A and B from data file */
Vscanf(" ' %ls '%*[\n] ", uplo);
if (*(unsigned char *)uplo == 'L')
    uplo_enum = Nag_Lower;
else if (*(unsigned char *)uplo == 'U')
    uplo_enum = Nag_Upper;
else
{
    Vprintf("Unrecognised character for Nag_UploType type\n");
    exit_status = -1;
    goto END;
}
if (uplo_enum == Nag_Upper)
{
    for (i = 1; i <= n; ++i)
    {
        for (j = i; j <= n; ++j)
            Vscanf("%lf", &A_UPPER(i,j));
        Vscanf("%*[\n] ");
    }
}
else
{
    for (i = 1; i <= n; ++i)
    {
        for (j = 1; j <= i; ++j)
            Vscanf("%lf", &A_LOWER(i,j));
        Vscanf("%*[\n] ");
    }
}
for (i = 1; i <= n; ++i)
{
    for (j = 1; j <= nrhs; ++j)
        Vscanf("%lf", &B(i,j));
}
Vscanf("%*[\n] ");

/* Compute solution */
/* nag_dtptrs (f07uec).
 * Solution of real triangular system of linear equations,
 * multiple right-hand sides, packed storage
 */
nag_dtptrs(order, uplo_enum, Nag_NoTrans, Nag_NonUnitDiag, n,
           nrhs, ap, b, pdb, &fail);
if (fail.code != NE_NOERROR)
{
    Vprintf("Error from nag_dtptrs (f07uec).\n%s\n", fail.message);
    exit_status = 1;
    goto END;
}
/* Print solution */
/* nag_gen_real_mat_print (x04cac).
```

```

    * Print real general matrix (easy-to-use)
    */
    nag_gen_real_mat_print(order, Nag_GeneralMatrix, Nag_NonUnitDiag, n, nrhs,
                           b, pdb, "Solution(s)", 0, &fail);
    if (fail.code != NE_NOERROR)
    {
        Vprintf("Error from nag_gen_real_mat_print (x04cac).\n%s\n",
                fail.message);
        exit_status = 1;
        goto END;
    }
END:
    if (ap) NAG_FREE(ap);
    if (b) NAG_FREE(b);

    return exit_status;
}

```

9.2 Program Data

```

nag_dtptrs (f07uec) Example Program Data
  4  2                               :Values of N and NRHS
  'L'                               :Value of UPLO
  4.30
 -3.96 -4.87
  0.40  0.31 -8.02
 -0.27  0.07 -5.95  0.12   :End of matrix A
-12.90 -21.50
 16.75 14.93
-17.55  6.33
-11.04  8.09               :End of matrix B

```

9.3 Program Results

nag_dtptrs (f07uec) Example Program Results

Solution(s)

	1	2
1	-3.0000	-5.0000
2	-1.0000	1.0000
3	2.0000	-1.0000
4	1.0000	6.0000
